A modified pattern matrix algorithm for multichain MDPs(The Development of Information and Decision Processes)

AUTHOR(S):

IKI, Tetsuichiro; HORIGUCHI, Masayuki

数理解析研究所講究録
1504巻 2006年 73-86

A modified pattern matrix algorithm for multichain MDPs

宮崎大学・教育文化学部　伊喜哲一郎 (Tetsuichiro IKI)
Faculty of Education and Culture, Miyazaki University
弓削商船高等専門学校・総合教育科　堀口正之 (Masayuki HORIGUCHI)
General Education, Yuge National College of Maritime Technology

Abstract

A pattern matrix algorithm for multichain Markov decision processes with average criteria proposed in our previous work[10] will be modified by use of the so-called vanishing discount approach by letting $\tau \to 0$ for the $(1-\tau)$ discount case.

Key words: multichain Markov decision processes, average reward structured algorithm, communicating class, transient class, value iteration, vanishing discount approach.

1 Introduction and notation

In this paper, we are concerned with the structured pattern matrix algorithm for multichain finite state Markov decision processes (MDPs) with an average reward criterion. The efficient algorithm for finding an optimal policy in average reward MDPs have been studied by many authors (cf. [5, 9, 13, 14, 15, 19]). For the unichain or communicating case, the optimal policy can be found by solving a single optimality equation (cf. [15]). However, for the multichain case, the optimal policy is characterized by two equations, called the multichain optimality equations, which are solved, for example, by linear programming (cf. [6, 15]) or policy iteration algorithms (cf. [7, 9, 15]).

The value iteration method based on classification of the state space into closed communicating subsets and transient one has been given by Schweitzer[17] and Ohno[14]. Ohno[14] has given the stopping rule to find an ε-optimal policy in a finite number iterations using by Schweitzer's value iteration method. Recently, Leizarowitz[13] has extended the above algorithm to the case of compact state space. In our previous work[10], we have proposed an algorithm for the multichain finite state MDPs in which the state classification is done by use of the corresponding pattern matrices and the idea of value iteration algorithm. However, the finding of an optimal policy for each communicating subset is supposed to use the policy improvement.

The objective of this paper is to propose the modified algorithm in which, to obtain an optimal policy for each communicating class, we use the so-called vanishing discount approach by considering the corresponding $(1-\tau)$ discounted expected reward as letting $\tau \to 0$.

In the reminder of this section, we will define finite state MDPs to be examined and describe the basic results for the average and discounted case.

We define a controlled dynamic system with a finite state space denoted by $S = \{1, 2, \ldots, N\}$. Associated with each state $i \in S$ is a non-empty finite set $A(i)$ of available actions. When the system is in state $i \in S$ and action $a \in A(i)$ is taken, then the system

moves to a new state $j \in S$ with probability $q_{ij}(a)$ where $\sum_{j\in S} q_{ij}(a) = 1$ for all $i \in S$ and $a \in A(i)$ and the reward $r(i,a)$ is earned. The process is repeated from the new state $j \in S$. This structure is called a Markov decision process, denoted by an MDP $\Gamma = (S, \{A(i) : i \in S\}, Q, r)$, where $Q = (q_{ij}(a) : i, j \in S, a \in A(i))$ and $r = r(i,a) \in \mathbb{R}$ for all $i \in S$ and $a \in A(i)$ and $\mathbb{R}$ is the set of all real numbers. The set of admissible state-action pairs will be denoted by

$$\mathbb{K} = \{(i,a) : i \in S, a \in A(i)\}.$$

The sample space is the product space $\Omega = \mathbb{K}^\infty$ such that the projection (X_t, Δ_t) on the t-th factor describes the state and action at the t-th time of the process ($t \geqq 0$). A policy $\pi = (\pi_0, \pi_1, \ldots)$ is a sequence of conditional probabilities with $\pi_t(A(x_t)|x_0, a_0, \ldots, x_t) = 1$ for all histories $(x_0, a_0, \ldots, x_t) \in \mathbb{K}^t S (t \geqq 0)$ where $\mathbb{K}^0 S = S$. The set of all policies is denoted by Π. A policy $\pi = (\pi_0, \pi_1, \ldots)$ is called randomized stationary if a conditional probability $\gamma = (\gamma(\cdot|i) : i \in S)$ given S exists, for which $\pi(\cdot|x_0, a_0, \ldots, x_t) = \gamma(\cdot|x_t)$ for all $t \geqq 0$ and $(x_0, a_0, \ldots, x_t) \in \mathbb{K}^t S$. Such a policy is simply denoted by γ. We denote by F the set of functions on S with $f(i) \in A(i)$ for all $i \in S$. A randomized stationary policy γ is called stationary if there exists a function $f \in F$ such that $\gamma(\{f(i)\}|i) = 1$ for all $i \in S$, which is denoted simply by f. For each $\pi \in \Pi$, starting state $X_0 = i$, the probability measure $P_\pi(\cdot|X_0 = i)$ on Ω is defined in a usual way. The problem we are concerned with is the maximization of the long-run expected average reward per unit time, which is defined by

$$\psi(i,\pi) = \liminf_{T\to\infty} \frac{1}{T} E_\pi \left(\sum_{t=0}^{T-1} r(X_t, \Delta_t) \,\middle|\, X_0 = i \right), \tag{1.1}$$

where $E_\pi(\cdot|X_0 = i)$ is the expectation w.r.t. $P_\pi(\cdot|X_0 = i)$.

A policy $\pi^* \in \Pi$ satisfying that

$$\psi(i,\pi^*) = \sup_{\pi\in\Pi} \psi(i,\pi) := \psi^*(i) \quad \text{for all } i \in S \tag{1.2}$$

is called to be average optimal or simply optimal.

The structured algorithm treated with in this paper is based on a communicating model introduced by Bather[1]. We say that the MDP Γ is communicating if there exists a randomized stationary policy $\gamma = (\gamma(\cdot|i) : i \in S)$ satisfying that the transition matrix $Q(\gamma)$ induced by γ defines a irreducible Markov chain where $Q(\gamma) = (q_{ij}(\gamma))$ with $q_{ij}(\gamma) = \sum_{a\in A(i)} q_{ij}(a)\gamma(a|i)$ for all $i, j \in S$. Let $B(S)$ be the set of all function on S. The following fact is well known.

Lemma 1.1 (cf. [15]). *Suppose that there exists a constant g and a $v \in B(S)$ such that*

$$v_i = \max_{a\in A(i)} \{r(i,a) + \sum_{j\in S} q_{ij}(a) v_j\} - g \quad \textit{for all } i \in S. \tag{1.3}$$

Then, g is unique and $g = \psi^*(i) = \psi^*(i,f)$ for all $i \in S$, where $f(i)$ is a maximizer in the right-hand side of (1.3) for all $i \in S$.

The expected total $(1-\tau)$-discounted reward is defined by

$$(1.4) \qquad v_\tau(i,\pi) = E_\pi\left(\sum_{t=0}^{\infty}(1-\tau)^t r(X_t,\Delta_t)\;\middle|\; X_0 = i\right) \quad \text{for } i \in S \text{ and } \pi \in \Pi,$$

and $v_\tau(i) = \sup_{\pi\in\Pi} v_\tau(i,\pi)$ is called a $(1-\tau)$-discounted value function, where $(1-\tau) \in (0,1)$ is a given discount factor.

For any $\tau \in (0,1)$, we define the operator $U_\tau : B(S) \to B(S)$ by

$$(1.5) \qquad U_\tau u(i) = \max_{a\in A}\left\{r(i,a) + (1-\tau)\sum_{j\in S} q_{ij}(a)u(j)\right\} \quad \text{for all } i \in S \text{ and } u \in B(S).$$

We have the following.

Lemma 1.2 ([15]). *It holds that*

(i) the operator U_τ is a contraction with the modulus $(1-\tau)$ and

(ii) the $(1-\tau)$-discount value function $v_\tau(i)$ is a unique fixed point of U_τ, i.e.,

$$(1.6) \qquad v_\tau = U_\tau v_\tau,$$

(iii) $v_\tau(i) = v_\tau(i, f_\tau)$ and $\lim_{\tau\to 0}\tau v_\tau(i) = \psi^*(i)$, where f_τ is a maximizer of the right-hand side in (1.6).

In Section 2, the methods of classifying the set of states by use of the corresponding pattern matrices is proposed, which is used to find an optimal policy by the value iteration algorithm in Section 4. In section 3, an optimal policy for each communicating class is obtained by use of the vanishing discount approach by letting $\tau \to 0$. In Section 5, a numerical example is given to comprehend our structured algorithm.

2 Classification of the states

In this section, from our previous work[10] we quote the method of classifying the state space and finding the maximum sub-MDPs by use of the corresponding pattern matrices, whose idea is essentially the same as [13, 17].

For a non-empty subset D of S, if, for each $i \in D$, there exists a non-empty subset $\overline{A}(i) \subset A(i)$ with $\sum_{j\in D} q_{ij}(a) = 1$ for all $a \in \overline{A}(i)$, we can consider the sub-MDP with the restricted state space D and available action space $\overline{A}(i)$ for $i \in D$, which is denoted by $\overline{\Gamma} = (D, \{\overline{A}(i) : i \in D\}, Q_D, r_D)$ where Q_D and r_D are restriction of Q and r on $\{(i,a) : i \in D, a \in \overline{A}(i)\}$. For any sub-MDP $\overline{\Gamma} = (D, \{\overline{A}(i) : i \in D\}, Q_D, r_D)$, a non-empty subset $\overline{D}$ of D is a communicating class in $\overline{\Gamma}$ if $\overline{D}$ is closed, that is, $\sum_{j\in\overline{D}} q_{ij}(a) = 1$ for all $i \in \overline{D}$ and $a \in \overline{A}(i)$ and the sub-MDP $(\overline{D}, \{\overline{A}(i) : i \in \overline{D}\}, Q_{\overline{D}}, r_{\overline{D}})$ is communicating. Also, $\overline{D}$ is maximum communicating class if it is not strictly contained in any communicating class.

For any positive integer l, let $\mathbb{C}^l$ and $\mathbb{C}^{l\times l}$ be the sets of l-dimensional row vectors and $l \times l$ matrices with $\{0,1\}$-valued elements, respectively. The sum $(+)$ and product $(\cdot)$ operators among elements in $\mathbb{C}^l$ and $\mathbb{C}^{l\times l}$ is defined by $a+b = \max\{a,b\}$ and $a \cdot b = \min\{a,b\}$ for $a, b \in \{0,1\}$.

For each $i \in S$ and $a \in A$, we define $m_i(a) = (m_{i1}(a), m_{i2}(a), \ldots, m_{iN}(a)) \in \mathbb{C}^N$ by

$$m_{ij}(a) = \begin{cases} 1 & \text{if } q_{ij}(a) > 0, \\ 0 & \text{if } q_{ij}(a) = 0, \end{cases} \quad (j \in S). \tag{2.1}$$

For a sub-MDP $\overline{\Gamma} = (D, \{\overline{A}(i) : i \in D\}, Q_D, r_D)$ with $D = \{i_1, i_2, \ldots, i_l\}$, we put $m_i(\overline{\Gamma}) = (m_{ii_1}(\overline{\Gamma}), m_{ii_2}(\overline{\Gamma}), \ldots, m_{ii_l}(\overline{\Gamma})) = \sum_{a\in\overline{A}(i)} m_i(a|\overline{\Gamma})$ $(i \in D)$, where $m_i(a|\overline{\Gamma}) = (m_{ii_1}(a), m_{ii_2}(a), \ldots, m_{ii_l}(a))$ $(i \in D, a \in \overline{A}(i))$. Using $m_i(\overline{\Gamma})$ $(i \in D)$, we define a pattern matrix $M(\overline{\Gamma})$ for the sub-MDP $\overline{\Gamma}$ by

$$M(\overline{\Gamma}) = \begin{pmatrix} m_{i_1}(\overline{\Gamma}) \\ \vdots \\ m_{i_l}(\overline{\Gamma}) \end{pmatrix} \in \mathbb{C}^{l\times l}. \tag{2.2}$$

We note that for $i, j \in D$, $m_{ij}(\overline{\Gamma}) = 1$ means that there exists $a \in \overline{A}(i)$ with $q_{ij}(a) > 0$.

The pattern matrix defined above is called a communication matrix (cf. [13]). However, since $M(\overline{\Gamma})$ determines the behaviour pattern of the sub-MDP $\overline{\Gamma}$, we call it a pattern matrix in this paper. For the pattern matrix $M(\overline{\Gamma})$, we define $\overline{M}(\overline{\Gamma}) \in \mathbb{C}^{l\times l}$ by

$$\overline{M}(\overline{\Gamma}) = \sum_{k=1}^{N} M(\overline{\Gamma})^k. \tag{2.3}$$

Then, we have the following.

Lemma 2.1. *For a non-empty subset $\overline{D}$ of D, $\overline{D}$ is a communicating class if and only if, for each $i \in \overline{D}$,*

$$\overline{m}_{ij}(\overline{\Gamma}) = \begin{cases} 1 & \text{if } j \in \overline{D}, \\ 0 & \text{if } j \notin \overline{D}, \end{cases}$$

where $\overline{m}_{ij}(\overline{\Gamma})$ is the (i,j) element of $\overline{M}(\overline{\Gamma})$ and $\overline{M}(\overline{\Gamma}) = (\overline{m}_{ij}(\overline{\Gamma}))$.

By reordering the states in D, $\overline{M}(\overline{\Gamma})$ can be transformed to the standard form:

$$\overline{M}(\overline{\Gamma}) = \begin{pmatrix} E_1 & & & \\ & E_2 & & 0 \\ & & \ddots & \\ 0 & & & E_d \\ & R & & K \end{pmatrix} \quad (d \geqq 1), \tag{2.4}$$

where E_j is a square matrix whose elements are all 1 $(1 \leqq j \leqq d)$ and $[R\ K]$ does not include a sub-matrix having the above form.

For any sets U, V, if $U \cap V = \emptyset$, we write the union $U \cup V$ by $U + V$.

Here, we get the classification of the state space.

Theorem 2.1. *For a sub-MDP* $\overline{\Gamma} = (D, \{\overline{A}(i) : i \in D\}, Q_D, r_D)$, *the state space* D *is classified as follows:*

$$D = U_1 + U_2 + \cdots + U_d + L \quad (d \geqq 1), \tag{2.5}$$

where U_j *is a communicating class for* $\overline{\Gamma}$ $(1 \leqq j \leqq d)$ *and* L *is not closed.*

The algorithm of obtaining the state-decomposition (2.5) through (2.4) by use of the pattern matrix will be called **Algorithm A**.

When the not-closed class L in (2.5) is non-empty, we go on with the state classification of L by finding the maximum sub-MDP. The basic idea of the following algorithm is the same as in [13], called **Algorithm B** hereafter.

Algorithm B:

1. Set $K_1 = L$ and $n = 1$.
2. Suppose that $\{K_i : 1 \leqq i \leqq n\}$ with $K_1 \supsetneqq K_2 \supsetneqq \cdots \supsetneqq K_n$ $(K_n \neq \emptyset)$ is given. Then, set $V = S - K_n$ and each $i \in K_n$, set $A_1(i) = A(i) - T(i)$, where $T(i) = \{a \in A(i) | \sum_{j \in V} m_{ij}(a) = 1\}$. Set $K_{n+1} = \{i \in K_n | A_1(i) \neq \emptyset\}$.
3. The following three cases happen:

Case 1: $K_{n+1} \neq \emptyset$ and $K_n \supsetneqq K_{n+1}$.
For this case, put $n = n + 1$ and go to Step 2 with $\{K_i : 1 \leqq i \leqq n + 1\}$.

Case 2: $K_{n+1} = K_n$.
For this case, set $\overline{D} := K_{n+1}$. Then, $\overline{\Gamma} = \{\overline{D}, \{A_1(i) : i \in \overline{D}\}, Q_{\overline{D}}, r_{\overline{D}}\}$ is a maximum sub-MDP in L. Apply **Algorithm A** for this sub-MDP $\overline{\Gamma}$.

Case 3: $K_{n+1} = \emptyset$. *Stop the algorithm.*
For this case, the set L does not include any sub-MDP and it holds that

$$\max_{i \in L, \pi \in \Pi} P_\pi(X_n \in L | X_0 = i) < 1. \tag{2.6}$$

Starting with the MDP $\Gamma = (S, \{A(i) : i \in S\}, Q, r)$ given firstly, we apply **Algorithm A** and **B** iteratively, so that we get the following.

Theorem 2.2. *Let* $\Gamma = (S, \{A(i) : i \in S\}, Q, r)$ *be the fixed MDP. Then, there exists a sequence of sub-MDPs*

$$\Gamma_k = (S_k, \{A_k(i) : i \in S_k\}, Q_{S_k}, r_{S_k}) \quad (k = 0, 1, \ldots, n^*)$$

satisfying the following (i)–(ii).

(i) $S_0 = S \supsetneqq S_1 \supsetneqq \cdots \supsetneqq S_{n^*}$

(ii) The state space S is decomposed to:

$$S = U_0 + U_1 + \cdots + U_{n^*} + L, \tag{2.7}$$

where

$$U_k = U_{k1} + U_{k2} + \cdots + U_{kl_k} \quad (0 \leqq k \leqq n^*), \tag{2.8}$$

U_{kj} $(1 \leqq j \leqq l_k)$ is a maximum communicating class (m.c.c.) for the sub-MDP Γ_k, and L is a transient class, that is, for any $i \in L$ and $a \in A(i)$, there exists an integer $n \geqq 1$ such that

$$\max_{i \in L, \pi \in \Pi} P_\pi(X_n \in L | X_0 = i) < 1. \tag{2.9}$$

3 Optimal policies for the communicating class

In this section, we show the method of finding an optimal policy for the communicating class by letting $\tau \to 0$ in the discount criterion case.

For the communicating case, we have the following.

Lemma 3.1 ([11]). *Suppose that $Q = (q_{ij}(a))$ is communicating. Then, there is a constant M such that*

$$\limsup_{\tau \to 0} \mid v_\tau(i) - v_\tau(j) \mid \leqq M \quad \textit{for all} \quad i, j \in S. \tag{3.1}$$

Let $P(S)$ be the set of all probability distributions on S, i.e.,

$$P(S) = \left\{ \mu = (\mu_1, \mu_2, \ldots, \mu_N) \mid \mu_i \geqq 0, \sum_{i=1}^{N} \mu_i = 1 \text{ for all } i \in S \right\}.$$

Let $Q = (q_{ij}(a))$. For any $\tau \in (0, 1)$ and $\mu = (\mu_1, \mu_2, \ldots, \mu_N) \in P(S)$, we perturb Q to $Q^{\tau,\mu} = \left(q_{ij}^{\tau,\mu}(a)\right)$ which is defined by

$$q_{ij}^{\tau,\mu}(a) = \tau\mu_j + (1 - \tau)q_{ij}(a) \quad \text{for } i, j \in S \text{ and } a \in A. \tag{3.2}$$

The matrix expression of (3.2) is $Q^{\tau,\mu} = \tau e\mu + (1 - \tau)Q$, where $e = (1, 1, \ldots, 1)^t$ is a transpose of N-dimensional vector $(1, 1, \ldots, 1)$. Then, we find that (1.6) in Lemma 1.2 can be rewritten as follows.

$$v_\tau(i) = \max_{a \in A} \left\{ r(i, a) + \sum_{j \in S} q_{ij}^{\mu,\tau}(a) v_\tau(j) \right\} - \tau \sum_{j \in S} \mu_j v_\tau(j) \quad \text{for all } i \in S. \tag{3.3}$$

For any fixed $i_0 \in S$, let $u_\tau(j) := v_\tau(j) - v_\tau(i_0)$ for all $j \in S$. Then, from (3.3), we get

$$u_\tau(i) = \max_{a \in A} \left\{ r(i, a) + \sum_{j \in S} q_{ij}^{\mu,\tau}(a) u_\tau(j) \right\} - \tau \sum_{j \in S} \mu_j v_\tau(j) \quad (i \in S). \tag{3.4}$$

From Lemma 3.1, for each $j \in S$, $u_\tau(j)$ is uniformly bounded and continuous in $\tau \in (0,1)$, so that we can imagine that $u_\tau \to u$ as $\tau \to 0$ for some $u \in B(S)$. Also, by Lemma 1.2 (iii), $\tau \sum_{j \in S} \mu_j v_\tau(j)$ in (3.3) converges to $\psi^* = \sum_{j \in S} \mu_j \psi^*(j)$. Thus, since $q_{ij}^{\mu,\tau}(a) \to q_{ij}(a)$ as $\tau \to 0$, we get the following average optimality equation:

$$u(i) = \max_{a \in A} \left\{ r(i,a) + \sum_{j \in S} q_{ij}(a) u(j) \right\} - \psi^* \quad (i \in S), \tag{3.5}$$

Observing that both S and A are supposed to be finite sets, for sufficiently small $\tau > 0$, we have that $f_\tau = f^*$, where f^* is a maximizer of the right-hand side of (3.5), which is average optimal.

From the above discussion, we have the following.

Theorem 3.1. *Suppose that $Q = (q_{ij}(a))$ is communicating. Then, it holds that*

(i) $\psi^*(i)(:= \psi^*)$ is independent of $i \in S$ and there exist a $u \in B(S)$ satisfying (3.5),

(ii) for any $\mu \in P(S)$, $\tau \sum_{j \in S} \mu_j v_\tau(j)$ in (3.3) converges to ψ^* as $\tau \to 0$, and

(iii) there exists $\tau_0 \in (0,1)$ such that f_τ in Lemma 1.2 (iii) is average optimal for any $\tau \in (0, \tau_0)$.

For sufficiently small $\tau > 0$, applying Theorem 3.1 to each communicating sub-MDP $\Gamma_{kj} = (U_{kj}, \{A_k(i) : i \in U_{kj}\}, Q_{U_{kj}}, r_{U_{kj}})$, we get an optimal stationary policy f_{kj} and a nearly optimal average reward g_{kj}^τ for sub-MDP Γ_{kj}, called relatively o.p. and relatively n.a.r., respectively, which is summarized in Table 1.
We note that a stationary policy f_{0j} is absolutely optimal on U_{0j} $(1 \leqq j \leqq l_0)$ because optimization is done in the MDP Γ.

4 Algorithm for obtaining an optimal policy

In this section, from[10] we review a value iterative algorithm based on data in Table 1 to find an (absolutely) optimal policy for the MDP Γ, and show the effectiveness of the algorithm.

Let $K_L := \{(i,a) | i \in L \text{ and } a \in A(i)\}$ and $B(L) = \{v | v : L \to \mathbb{R}\}$. For any function d on K_L, the map $T_d : B(L) \to B(L)$ will be defined as

$$T_d v(i) = \max_{a \in A(i)} \{ d(i,a) + \sum_{j \in L} q_{ij}(a) v(j) \} \quad (i \in L). \tag{4.1}$$

The map T_d is shown to be monotone and n-step contractive(cf. [1, 10]) where n is given in (2.9). For each function d on K_L, the unique fixed point of T_d will be denoted by $v\{d\}$.

Let $\mathcal{K} = \{(s,j) | 0 \leqq s \leqq n^*, 1 \leqq j \leqq l_s\}$ where n^* and l_s are given in Theorem 2.2. For $D \subset S$, let $q_i(D|a) := \sum_{j \in D} q_{ij}(a)$. In the ensuring discussion, we give the value iteration algorithm, called **Algorithm $\mathbf{C}^\tau$**, with data in Table 1.

Algorithm $\mathbf{C}^\tau$.

1. Set $n = 1, g_{sj}(1) = g^{\tau}_{sj}$ for $(s,j) \in \mathcal{K}$ and $g_i(1) = v\{d_1\}(i)$ for $i \in L$, where $d_1(i,a) = \sum_{(s,j)\in\mathcal{K}} q_i(U_{sj}|a) g_{sj}(1)$.

2. Suppose that $\{g_{sj}(n) : (s,j) \in \mathcal{K}\}$ and $\{g_i(n) : i \in L\}$ are given $(n \geqq 1)$. Let for $i \in S - L$,

$$g_i = \max_{a\in A(i)} \{d_n(i,a) + \sum_{j\in L} q_{ij}(a) g_j(n)\}, \tag{4.2}$$

where $d_n(i,a) = \sum_{(s,j)\in\mathcal{K}} q_i(U_{sj}|a) g_{sj}(n)$.
Let $g_{sj}(n+1) = \max_{i\in U_{sj}} g_i$ and $g_i(n+1) = v\{d_n\}(i)$ for $i \in L$.

3. Let $n = n + 1$ and go to Step 2.

Concerning with **Algorithm** $\mathbf{C}^{\tau}$, we have the following.

Lemma 4.1 ([10]). *In* **Algorithm** $\mathbf{C}^{\tau}$, *we have:*

(i) It holds that

$$g_{sj}(n+1) \geqq g_{sj}(n) \text{ for } (s,j) \in \mathcal{K}, \text{ and} \tag{4.3}$$

$$g_i(n+1) \geqq g_i(n) \text{ for } i \in L. \tag{4.4}$$

(ii) The **Algorithm** $\mathbf{C}^{\tau}$ *converges, i.e., when* $n \to \infty$ $g_{sj}(n) \to \overline{g}_{sj}$ *for* $(s,j) \in \mathcal{K}$ *and* $g_i(n) \to \overline{g}_i$ *for* $i \in L$.

For $\overline{g}_{sj}$ and $\overline{g}_i$ in Lemma 4.1, we have the following.

$$\overline{g}_{sj} = \max_{a\in A(i)} \{d(i,a) + \sum_{j\in L} q_{ij}(a)\overline{g}_j\} \text{ for } i \in U_{sj} \text{ and } (s,j) \in \mathcal{K}, \tag{4.5}$$

and

$$\overline{g}_i = \max_{a\in A(i)} \{d(i,a) + \sum_{j\in L} q_{ij}(a)\overline{g}_j\} \text{ for } i \in L, \tag{4.6}$$

where $d(i,a) = \sum_{(s,j)\in\mathcal{K}} q_i(U_{sj}|a)\overline{g}_{sj}$.

Let f^* be any stationary policy such that

$$f^*(i) = \begin{cases} f_{0j}(i) & \text{for } i \in U_{0j} \ (1 \leqq j \leqq l_0) \\ \text{any maximizer in (4.5) and (4.6)} & \text{for } i \in S_1. \end{cases} \tag{4.7}$$

Since $\overline{g}_{sj}, \overline{g}_i$ and f^* in (4.5)–(4.7) are depending on $\tau \in (0,1)$, we denote them by $\overline{g}^{\tau}_{sj}, \overline{g}^{\tau}_i$ and f^*_{τ} respectively. Then, we have the following from Theorem 3.1.

Theorem 4.1. *It holds that*

(i) there exists $\tau_0 \in (0,1)$ *such that* f^*_{τ} *is average optimal for any* $\tau (0 < \tau < \tau_0)$, *and*

(ii) as $\tau \to 0, \overline{g}^{\tau}_{sj} \to \psi^*(i)$ *for* $i \in U_{sj}, (s,j) \in \mathcal{K}$, *and* $\overline{g}^{\tau}_i \to \psi^*(i)$ *for* $i \in L$.

5 A numerical example

Here, in order to comprehend our modified algorithms working effectively we will consider a numerical example as follows, which is dealt with in [10]. In our previous work[10], the finding of an optimal policy for each communicating subset is supposed to use the policy improvement. In this paper, we use the nearly optimal policy $\overline{f}_\tau$ and nearly optimal average reward $\overline{g}^\tau_{sj}, (s,j) \in \mathcal{K}$ in each communicating sub-MDPs with sufficiently small $\tau > 0$.

Let $S = \{1,2,3,4,5,6,7,8\}$ and $A(1) = \{1,2\}, A(2) = \{1\}, A(3) = \{1,2,3\}, A(4) = \{1,2\}, A(5) = \{1,2\}, A(6) = \{1,2,3\}, A(7) = \{1,2\}$ and $A(8) = \{1,2,3\}$, whose transition probability matrix $Q = (q_{ij}(a))$ and rewards $r = r(i,a), i,j \in S, a \in A(i)$ are given in Table 2.

First, applying the **Algorithm A** we have the classification of the states:

$$S = U_{01} + U_{02} + L,$$

where $U_{01} = \{3,6,8\}, U_{02} = \{2,4\}, L = \{1,5,7\}$. Next, we apply **Algorithm B** to the set of transient states $L = \{1,5,7\}$. Then, we find the maximum sub-MDP

$$\Gamma_1 = (S_1, \{A_1(i), i \in S_1\}, Q_{S_1}, r_{S_1})$$

where $S_1 = \{5,7\}, A_1(5) = \{1\}, A_1(7) = \{3\}$ and Q_{S_1}, r_{S_1} are restrictions of Q, r on $i,j \in S_1$ and $a \in A(i_1), i_1 \in S_1$, respectively. Applying **Algorithm A** to Γ_1, we find that Γ_1 is communicating and hence we set $U_{11} = \{5,7\}$. In the end, the decomposition of S in (2.7) is shown as

$$S = U_{01} + U_{02} + U_{11} + L \text{ with } L = \{1\}.$$

Next, for each communicating class we calculate a nearly optimal policy(n.o.p., for short) and nearly optimal average reward(n.a.r., for short) through the vanishing discount approach. We set discount rate $\tau = 0.0001$ and $g^\tau_{sj}, (s,j) \in \mathcal{K}$ are replaced by arithmetic mean of $\tau\overline{v}_\tau(i), i \in U_{sj}$ where $\tau\overline{v}_\tau(i)$ is n.a.r. by value iteration with repeating the steps n until $\max_{i \in U_{sj}} |\overline{v}^{n+1}_\tau(i) - \overline{v}^n_\tau(i)| < \varepsilon := 10^{-6}$. Then, data in Table 1 is given in Table 3.

Applying **Algorithm C**$^\tau$ to Table 3, we have n.o.p. and n.a.r. as in Table 4. Moreover, if we use the policy iteration algorithm for each m.c.c. in order to get relatively a.r., then applying **Algorithm C** in [10], the optimal policy(o.p., for short) and optimal average rewards(o.a.p., for short) can be found as in Table 4 such that

$$f^*(3) = f^*(6) = f^*(8) = 2, f^*(2) = 1, f^*(4) = 2, f^*(5) = 1, f^*(7) = 3, f^*(1) = 2$$

and

$$\psi^*(3) = \psi^*(6) = \psi^*(8) = 11.333333, \psi^*(2) = \psi^*(4) = 9.714286,$$
$$\psi^*(5) = \psi^*(7) \cong 10.793648, \psi^*(1) \cong 10.793650.$$

It is noted that in **Algorithm C**$^\tau$, the process are lumped as Table 5. On the other hand, by using the iterative method([8]) with the same rule $\varepsilon = 10^{-6}$ for repeating the algorithm we have the following result as in Table 6 through **Algorithm C** in [10].

While the algorithm in the iterative method([8]) calculates nearly optimal average rewards directly, the vanishing discount approach calculates nearly optimal discount reward $\overline{v}_\tau$ firstly, and then n.a.r. is given by $\tau\overline{v}_\tau$. Hence, for sufficiently small $\tau > 0$, vanishing discount approach can get more significant digits which is close to the optimal value than the method in [8] if we repeat the value iteration algorithm until $\max_{i\in U_{sj}} |\overline{v}_\tau^{n+1}(i) - \overline{v}_\tau^{n}(i)| < \varepsilon$. Moreover, our modified algorithm only use value iteration method, so that it is easily to calculate the nearly optimal values with smaller numbers of multiplication per iteration than other algorithms. Table 7 shows the results of applying the **Algorithm C**$^\tau$ for some cases of τ. Table 4 illustrates that our modified algorithm works well in this example.

References

[1] John Bather. Optimal decision procedures for finite Markov chains. II. Communicating systems. *Advances in Appl. Probability*, 5:521–540, 1973.

[2] Richard Bellman. *Dynamic programming.* Princeton Univeristy Press, Princeton, N. J., 1957.

[3] Eric V. Denardo. Contraction mappings in the theory underlying dynamic programming. *SIAM Rev.*, 9:165–177, 1967.

[4] Eric V. Denardo. *Dynamic programming: models and applications.* Prentice-Hall, Inc., Englewood Cliffs, N. J., 1982.

[5] A. Federgruen and P. J. Schweitzer. Discounted and undiscounted value-iteration in Markov decision problems: a survey. In *Dynamic programming and its applications (Proc. Conf., Univ. British Columbia, Vancouver, B.C., 1977)*, pages 23–52. Academic Press, New York, 1978.

[6] A. Hordijk and L. C. M. Kallenberg. Linear programming and Markov decision chains. *Management Sci.*, 25(4):352–362, 1979/80.

[7] Arie Hordijk and Martin L. Puterman. On the convergence of policy iteration in finite state undiscounted Markov decision processes: the unichain case. *Math. Oper. Res.*, 12(1):163–176, 1987.

[8] Arie Hordijk and Henk Tijms. A modified form of the iterative method of dynamic programming. *Ann. Statist.*, 3:203–208, 1975.

[9] Ronald A. Howard. *Dynamic programming and Markov processes.* The Technology Press of M.I.T., Cambridge, Mass., 1960.

[10] T. Iki, M. Horiguchi, and M. Kurano. A structured pattern matrix algorithm for multichain markov decision processes. *(preprint)*, 2005.

[11] T. Iki, M. Horiguchi, M. Yasuda, and M. Kurano. A learning algorithm for communicating markov decision processes with unknown transition matrices. *(preprint)*, 2005.

[12] John G. Kemeny and J. Laurie Snell. *Finite Markov chains.* The University Series in Undergraduate Mathematics. D. Van Nostrand Co., Inc., Princeton, N.J.-Toronto-London-New York, 1960.

[13] Arie Leizarowitz. An algorithm to identify and compute average optimal policies in multichain Markov decision processes. *Math. Oper. Res.*, 28(3):553–586, 2003.

[14] K. Ohno. A value iteration method for undiscounted multichain Markov decision processes. *Z. Oper. Res.*, 32(2):71–93, 1988.

[15] Martin L. Puterman. *Markov decision processes: discrete stochastic dynamic programming.* John Wiley & Sons Inc., New York, 1994. A Wiley-Interscience Publication.

[16] P. J. Schweitzer. Iterative solution of the functional equations of undiscounted Markov renewal programming. *J. Math. Anal. Appl.*, 34:495–501, 1971.

[17] P. J. Schweitzer. A value-iteration scheme for undiscounted multichain Markov renewal programs. *Z. Oper. Res. Ser. A-B*, 28(5):A143–A152, 1984.

[18] E. Seneta. *Nonnegative matrices and Markov chains.* Springer Series in Statistics. Springer-Verlag, New York, second edition, 1981.

[19] D. J. White. Dynamic programming, Markov chains, and the method of successive approximations. *J. Math. Anal. Appl.*, 6:373–376, 1963.

m.c.c.	$U_{01},\ldots,U_{0l_0}$	$U_{11},\ldots,U_{1l_1}$	$\cdots$	$U_{n^*1},\ldots,U_{n^*l_{n^*}}$
relatively o.p.	$f_{01},\ldots,f_{0l_0}$	$f_{11},\ldots,f_{1l_1}$	$\cdots$	$f_{n^*1},\ldots,f_{n^*l_{n^*}}$
relatively n.a.r.	$g^T_{01},\ldots,g^T_{0l_0}$	$g^T_{11},\ldots,g^T_{1l_1}$	$\cdots$	$g^T_{n^*1},\ldots,g^T_{n^*l_{n^*}}$

Table 1: Relatively o.p. and n.a.r.

state	action	transition probabilities $q_{ij}(a)$								reward
i	$a \in A(i)$	$j=1$	$j=2$	$j=3$	$j=4$	$j=5$	$j=6$	$j=7$	$j=8$	$r(i,a)$
1	1	0	1/2	1/4	0	0	1/4	0	0	9
	2	1/8	0	0	1/4	0	1/4	1/8	1/4	10
2	1	0	7/10	0	3/10	0	0	0	0	11
3	1	0	0	1/2	0	0	1/4	0	1/4	5
	2	0	0	1/8	0	0	1/8	0	3/4	2
	3	0	0	3/16	0	0	3/4	0	1/16	2.5
4	1	0	3/5	0	2/5	0	0	0	0	7
	2	0	2/5	0	3/5	0	0	0	0	8
5	1	0	0	0	0	1/2	0	1/2	0	12
	2	1/8	1/2	0	0	1/8	1/8	1/8	0	2
6	1	0	0	1/4	0	0	1/2	0	1/4	6
	2	0	0	1/16	0	0	3/16	0	3/4	0.75
	3	0	0	5/8	0	0	1/4	0	1/8	2.25
7	1	1/4	0	0	0	1/4	0	1/2	0	6
	2	1/4	0	0	1/8	1/4	0	1/4	1/8	7
	3	0	0	0	0	1/4	0	3/4	0	10
8	1	0	0	1/2	0	0	1/2	0	0	14
	2	0	0	1/16	0	0	1/16	0	7/8	13

Table 2: A numerical example

m.c.c.	$U_{01} = \{3,6,8\}$	$U_{02} = \{2,4\}$	$U_{11} = \{5,7\}$
relatively n.o.p.	$\overline{f}(3) = \overline{f}(6) = \overline{f}(8) = 2$	$\overline{f}(2) = 1, \overline{f}(4) = 2$	$\overline{f}(5) = 1, \overline{f}(7) = 3$
relatively n.a.r.	11.332634	9.714254	10.666710

($\overline{f}_{sj}$ is abbreviated to $\overline{f}$)

Table 3: Relatively n.o.p. and n.a.r. for each maximum communicating subclass

m.c.c. and L	$U_{01} = \{3,6,8\}$	$U_{02} = \{2,4\}$	$U_{11} = \{5,7\}$	$L = \{1\}$
n.o.p.	$\overline{f}(3) = \overline{f}(6) = \overline{f}(8) = 2$	$\overline{f}(2) = 1, \overline{f}(4) = 2$	$\overline{f}(5) = 1, \overline{f}(7) = 3$	—
n.a.r.	11.332634	9.714254	10.793171	10.793171
o.p.	$f^*(3) = f^*(6) = f^*(8) = 2$	$f^*(2) = 1, f^*(4) = 2$	$f^*(5) = 1, f^*(7) = 3$	$f^*(1) = 2$
o.a.r.	11.333333	9.714286	10.793648	10.793650

Table 4: The table of n.o.p. and n.a.r. with $\tau = 0.0001$, and o.p. and o.a.r.

	i	a	U_{01}	U_{02}	U_{10}	L
U_{01}	3	1	1	0	0	0
		2	1	0	0	0
		3	1	0	0	0
	6	1	1	0	0	0
		2	1	0	0	0
		3	1	0	0	0
	8	1	1	0	0	0
		2	1	0	0	0
U_{02}	2	1	0	1	0	0
	4	1	0	1	0	0
		2	0	1	0	0
U_{10}	5	1	0	0	1	0
		2	1/8	1/2	1/4	1/8
	7	1	0	0	3/4	1/4
		2	1/8	1/8	1/2	1/4
		3	0	0	1	0
L	1	1	1/2	1/2	0	0
		2	1/2	1/4	1/8	1/8

Table 5: Lumped transition matrix

m.c.c. and L	$U_{01} = \{3, 6, 8\}$	$U_{02} = \{2, 4\}$	$U_{11} = \{5, 7\}$	$L = \{1\}$
relatively n.o.p.	$\overline{f}(3) = \overline{f}(6) = \overline{f}(8) = 2$	$\overline{f}(2) = 1, \overline{f}(4) = 2$	$\overline{f}(5) = 1, \overline{f}(7) = 3$	—
relatively n.a.r.	11.330055	9.715645	10.667990	—
n.o.p.	$\overline{f}(3) = \overline{f}(6) = \overline{f}(8) = 2$	$\overline{f}(2) = 1, \overline{f}(4) = 2$	$\overline{f}(5) = 1, \overline{f}(7) = 3$	—
n.a.r.	11.330055	9.715645	10.791915	10.791918

Table 6: The table of n.o.p. and n.a.r. with $\tau = 0.0001$ by using the iterative method in [8].

		$\tau = 0.01$		$\tau = 0.001$		$\tau = 0.0001$		$\tau = 0.00001$	
		n.o.p	n.a.r.	n.o.p	n.a.r.	n.o.p	n.a.r.	n.o.p	n.a.r.
U_{01}	3	2	11.225940	2	11.322578	2	11.332257	2	11.333225
	6	2	11.212615	2	11.321245	2	11.332123	2	11.333211
	8	2	11.352416	2	11.335243	2	11.333523	2	11.333351
ψ_τ^*			11.263657		11.326355		11.332634		11.333262
U_{02}	2	1	9.732574	1	9.716210	1	9.714468	1	9.714303
	4	2	9.689899	2	9.711837	2	9.714040	2	9.714260
ψ_τ^*			9.711237		9.714024		9.714254		9.714282
U_{11}	5	1	10.684384	1	10.668443	1	10.666843	1	10.666683
	7	1	10.657806	1	10.665777	1	10.666577	1	10.666657
ψ_τ^*			10.671095		10.667110		10.666710		10.666670

Table 7: Table of the results for applying the Algorithm C^τ